APPLICATION FOR UNITED STATES PATENT

in the name of

Alfonso Valdes, Ketih Skinner and Phillip Andrew Porras

of

SRI International, Inc.

for

Sensor and Alert Correlation in Intrusion Detection Systems

Kenneth F. Kozik
Fish & Richardson P.C.
225 Franklin Street
Boston, MA 02110-2804
Tel.: (617) 542-5070
Fax: (617) 542-8906

ATTORNEY DOCKET: **BOS-10454-KFK110900-1**

DATE OF DEPOSIT: November 9, 2000

EXPRESS MAIL NO.: EL 624 275 981 US

Sensor and Alert Correlation in Intrusion Detection Systems

Reference to Government Funding

This invention was made with Government support under contract number F30602-99-C-0149 awarded by the Air Force Research Laboratory. The Government has certain rights in this invention.

Related Application

This is a continuation-in-part of U.S. patent application serial number 09/653,066, filed September 1, 2000 and entitled "Methods for Detecting and Diagnosing Abnormalities Using Real-Time Bayes Networks," which is incorporated herein by reference.

Reference to Related Documents

This invention incorporates by reference a paper entitled "Probabilistic Approaches to Alert Management," by A. Valdes, attached as an appendix.

Background

The invention relates generally to intrusion detection, and more specifically to sensor and alert correlation in intrusion detection systems.

There are three main types of intrusion detection systems currently used to detect hacker attacks on computer networks: signature analysis systems, statistical analysis systems, and systems based on probabilistic reasoning. Signature analysis systems compare current data traffic patterns with stored traffic patterns representing the signature or profile of various types of hacker attacks. These systems generate an alert if the pattern of traffic received by the network matches one of the stored attack patterns.

Statistical analysis systems compare current data traffic patterns with statistical profiles of previous traffic patterns. These systems generate an alert if a current traffic pattern is significantly different from a stored profile of "normal" traffic. Examples of both statistical- and signature-based intrusion detection systems are described in Porras, et al., "*Live Traffic Analysis of TCP/IP Gateways*," Internet Society's Networks and Distributed Systems Society Symposium, March 1998.

Examples of intrusion detection systems based on probabilistic reasoning are described in U.S. patent application serial number 09/653,066 entitled "Methods for Detecting and Diagnosing Abnormalities Using Real-Time Bayes Networks." In one such system, a Bayes network is established that includes models (called "hypotheses") that represent both normal traffic and attack traffic received by a computer network. Traffic actually received by the network is examined in real-time to identify its relevant characteristics or features, such as volume of data transfer, number of erroneous connection requests, nature of erroneous connection requests, ports to which connections are attempted, etc. Information about these relevant features is then provided to the Bayes network, which calculates a system belief (a probability) that the current network traffic is either normal traffic or attack traffic.

A typical intrusion detection system may include one or more sensors that monitor network traffic in the manner discussed above, and one or more other sensors that monitor network resources. A system operator or network administrator (usually a person) reviews all of the alerts generated by the system.

A major problem with existing intrusion detection systems is that they often provide misleading, incomplete, or low-quality information to the system operator; this may make it impossible to take proper steps to protect the network. For example, during a large-scale hacker attack each of the system's sensors may generate hundreds of alerts. Although each alert may be accurate, the sheer number of alerts could easily overwhelm the system operator. Moreover, false alarms can be triggered by normal traffic directed towards a malfunctioning network resource, such as a server. These false alarms could distract the system operator from alerts triggered by actual hacker attacks. Finally, most intrusion detection systems are unable to detect low-level attacks such as port sweeps, in which hackers slowly "probe" a network to discover its structure and weaknesses.

Summary

This invention uses probabilistic correlation techniques to increase sensitivity, reduce false alarms, and improve alert report quality in intrusion detection systems. In one preferred embodiment, an intrusion detection system includes at least two sensors to monitor different aspects of a computer network, such as a sensor that monitors network

traffic and a sensor that discovers and monitors available network resources. The sensors are correlated in that the belief state of one sensor is used to update or modify the belief state of another sensor. For example, a network resource sensor may detect that a server is malfunctioning. This information is used to modify the belief state of one or more network traffic sensors so that normal network traffic directed towards the malfunctioning server does not trigger a false alarm.

In another example, a network resource sensor transmits its knowledge of network structure to a network traffic sensor. This information is used to modify the belief state of the network traffic sensor so that an attempt to communicate with a non-existent resource appears to be suspicious. By allowing different sensors to share information, the system's sensitivity to low-level attacks can be greatly increased, while at the same time greatly reducing the number of false alarms.

In another embodiment of this invention, probabilistic correlation techniques are used to organize alerts generated by different types of sensors. By comparing features of each new alert with features of previous alerts, and adjusting the comparison by an expectation that certain feature values will or will not match, the alerts can be grouped in an intelligent manner. The system operator may then be presented with a few groups of related alerts, rather than dozens of individual alerts from multiple sensors.

Brief description of the drawings

Figure 1 shows a preferred intrusion detection system with coupled sensors.

Figure 2 is a flowchart showing a preferred method for coupling sensors.

Figure 3 shows a preferred intrusion detection system with coupled sensors and an alert correlation device.

Figure 4 is a flowchart showing a preferred method for grouping alerts into classes of related alerts.

Detailed Description

I. Introduction

In preferred embodiments, intrusion detection systems for computer networks include sensors that monitor both network traffic and network resources. Correlation techniques are used to increase system sensitivity, reduce false alarms, and organize alerts into related groups. The sensor and alert correlation techniques described below may be used in intrusion detection systems that include any type or mix of sensors. However, probabilistic sensors similar to those described in U.S. Patent Application Serial Number 09/653,066 may provide advantages over other types of sensors.

Bayes networks are preferably used to perform the probabilistic reasoning functions described below. As is more fully described in U.S. Patent Application Serial Number 09/653,066, Bayes networks include models (called "hypotheses") that represent some condition or state; probabilistic reasoning methods are used to determine a belief that a current observation corresponds to one of the stored hypotheses. Before a Bayes network performs an analysis, an initial belief (called a "prior probability") regarding the various hypotheses is established. During the analysis, a new belief based on actual observations is established, and the prior probability is adjusted accordingly. In the context of the embodiments described here, a relevant Bayes hypothesis might be "there is a stealthy portsweep attack against the computer network." Belief in the portsweep attack hypothesis would be strengthened if attempts to connect to several invalid or nonexistent ports are observed.

II. Intrusion Detection with Correlated Sensors

As was discussed above, intrusion detection systems typically have several independent sensors that gather different types of information. In preferred embodiments of this invention, the information gathered by the various sensors may be shared to improve overall performance of the intrusion detection system.

In a preferred intrusion detection system 100 shown in Figure 1, sensors are coupled so that the belief state of one sensor (such as network resource sensor 103) affects the belief state of another sensor (such as network traffic sensor 101). Each

sensor may then transmit alerts to a system operator or network administrator 105. These sensors need not be probabilistic sensors; for example, a sensor that keeps track of the number of servers in a network has wouldn't need to generate a probabilistic output.

Figure 2 illustrates a preferred method by which sensors are coupled or correlated. At step 201, a first (preferably probabilistic) sensor receives all or part of the belief state of a second (not necessarily probabilistic) sensor. The belief state of the second sensor may indicate an apparent normal, degraded, or compromised state of a monitored system resource, the existence or validity of supported services, or any other relevant belief state held by a sensor in an intrusion detection system. At step 203, a prior belief state of the first sensor is adjusted, the adjustment based at least in part on the second sensor's belief state. For example, the prior belief state of the first sensor or may be adjusted so that an erroneous transaction with a damaged or compromised network resource does not generate an alert. In another example, the prior belief state of the first sensor may be adjusted so that an attempted communication with a system server or resource that does not exist appears to be suspicious. By allowing different sensors to share information, the system's sensitivity to low-level attacks can be greatly increased, while at the same time greatly reducing the number of false alarms.

III. Alert Correlation

In another embodiment of this invention, probabilistic correlation techniques are used to organize alerts into classes or groups of related alerts. By comparing features of each new alert with features of previous alerts, and adjusting the comparison by an expectation that certain feature values will or will not match, the alerts can be grouped in an intelligent manner. Alerts may be grouped or organized in several different ways, depending on both the type of an attack and the structures of the intrusion detection system and the monitored network. For example, all alerts related to a specific attack (such as a denial of service attack) may be grouped together. Or, all alerts related to the various stages of a staged attack may be grouped together to allow the system operator to view the progression of a staged attack.

A. Alert Correlation Devices

Figure 3 shows a preferred intrusion detection system 300 that includes a network traffic sensor 301 and network resource sensor 303. Intrusion detection system 300 may include any number of sensors, and the belief states of the sensors may be correlated in the manner discussed above. Alerts generated by sensors 301 and 303 are provided to an alert correlation device 305, which uses probabilistic reasoning (preferably Baysian) techniques to organize alerts into classes of related alerts. These alert classes are then provided to a system operator or network administrator 307.

B. Measurement of Similarity

To group alerts in an intelligent manner, it is necessary to define and determine the degree of similarity between a new alert and an existing alert or alert class. This measurement of similarity is preferably calculated by comparing similarity among shared features (also called "measures") of the alert and alert class. Examples of features to consider when assessing similarity between a new alert and an existing alert class include source IP address, destination IP address and port, type of alert, type of attack, etc. The nature of an alert may change the expectation of which features should be similar. It is therefore important to define a measurement of similarity to take into account which features are candidates for matching.

In the following discussion, X is defined to be the value of a feature or measure in a new alert, and Y is the value of that feature or measure in an existing alert class to which it is being compared. For features that have a single value, the features of a new alert and an existing alert class are defined to be similar if their feature values are equal. That is:

$$SIM(X,Y) = \begin{cases} 1.0, X = Y \\ 0, otherwise \end{cases}$$

The more general case is one in which a feature of an alert includes a list of observed values, and a "hit count" of the number of times each value has been observed. For reasons of normalization, the hit count may be converted to a probability. In this sense, X and Y are lists of feature and probability values, possibly of different lengths. A probability vector describes a pattern over observed categories, where:

$p_X(C)$ = probability of category C in list X,

$p_Y(C)$ = probability of category C in list Y,

$\mathbf{P}_X$ = probability vector over categories observed for X,

$\mathbf{P}_Y$ = probability vector over categories observed for Y.

The notation $C \in X$ is used to denote that category C occurs in list X. The similarity of the two lists is given by:

$$Sim(X,Y) = \frac{\left[\sum_{C \in X \text{ AND } C \in Y} P_X(C) \times P_Y(C)\right]^2}{(\mathbf{P}_X \bullet \mathbf{P}_X)(\mathbf{P}_Y \bullet \mathbf{P}_Y)}$$

If the two lists are the same length, then this measure gives the square of the cosine between the two. This has an intuitive geometric property that is not shared by, say, the dot product as is commonly used in the pattern matching literature.

If the patterns (1, 0), (0, 1), and (0.5, 0.5) are over the same two features, then:

$$Sim(\{1,0\},\{0,1\}) = 0$$

$$Sim(\{0.5,0.5\},\{0,1\}) = Sim(\{0.5,0.5\},\{1,0\}) = 0.5$$

In other words, the first two patterns are orthogonal, and the third is halfway in between.

C. Expectation of Similarity

As was discussed above, the nature of an alert may change an expectation of which features of a new alert and an existing alert class should be similar. For example, a syn flood is a type of attack in which the source IP address is typically forged. In this case, similarity in the source IP address would not be considered when assessing the overall similarity between a new alert triggered by a syn flood attack and an existing alert class.

In a preferred embodiment, the expectation of similarity is a feature-specific number between 0 and 1, with 1 indicating a strong expectation of a match, and 0 indicating that two otherwise similar alerts are likely not similar with respect to a specific feature.

An alert state is preferably represented as a probability vector over likely alert states. For each candidate alert state, each feature has a vector of the same length whose elements are the similarity expectation values (that is, numbers between 0 and 1) given

the state. These expectation values may initially be assigned a value of about 0.6 to indicate a medium expectation of similarity, except where it is known that the expectation of similarity is either lower or higher. For example, in an access from a compromised host or a denial of service (DOS) attack, the attacker often spoofs (makes up) the source IP address. Therefore, the expectation values may initially be assigned a value of about 0.1, indicating a low expectation that the source IP address will match that of the attacker.

Based on the alert state, the similarity expectation may be dynamically generated as the weighted sum of the elements of an expectation table, with the weights from the (evolving) alert state distribution. This is itself an expectation in the statistical sense, conditioned on the belief over the present alert state. The similarity expectation is therefore general enough to cover the situation of an unambiguous alert or "call" from a signature-based sensor, in which case the distribution over states is 1.0 at the state corresponding to the alert or call and 0 elsewhere. For example, algebraically for a feature or measure J:

$$E_J = \sum_{i \in \{alert\_states\}} BEL(i) E_J(i)$$

E_J = Expected similarity for measure J, given present state.

$BEL(i)$ =Belief that the attack state is presently i.

$E_J(i)$= Element i of the lookup table of expected similarity for measure J given state i.

Both the new alert and the alert class to which it is being compared each compute the similarity expectation for each feature. Feature similarity and similarity expectation are then combined to form a single value of alert similarity. This is done by combining feature similarities and normalizing by similarity expectation, over the set of common features. For example:

$$SIM(\mathbf{X},\mathbf{Y}) = \frac{\sum_J E_J^{\mathbf{X}} E_J^{\mathbf{Y}} SIM(X_J, Y_J)}{\sum_J E_J^{\mathbf{X}} E_J^{\mathbf{Y}}}$$

X =New alert

Y =Existing alert

J indexes measures

$E_J^{\mathbf{X(Y)}}$ = Similarity expectation for measure J, alert **X(Y)**.

A similar definition can be formed by taking products rather than sums above, and then taking the geometric mean. This has some advantages, but can be overly influenced by a single extremely small value.

Where appropriate, similarity expectation may also cover list containment and subnet similarity. For example, an intrusion detection system may generate two alerts, one from a network sensor and one from a host sensor. To decide if the target addresses of these two alerts are similar, the expectation of similarity might be that the target address in the alert generated by the host sensor would be contained in the list of target addresses in the network sensor's alert. In another example, an intermediate expectation of similarity may be generated if target addresses from two alerts do not match but appear to be from the same subnet.

D. Transition Models

When a new alert is encountered, all existing alert classes are preferably passed through their transition models to generate new prior belief states. Transition models attempt to assign a "time value" to alert confidence, as well as try to anticipate the next step in a staged attack. The "time value" is typically modeled as a multiplicative decay factor that reduces alert model confidence slowly over a period of days. The decay period varies by type of attack, and eventually decreases to some background level. This is used to downweight very old alert classes when there are multiple candidates to be compared with a new alert. Transition in time tends to decrease overall suspicion for a specific alert class, as well as spread suspicion over other hypotheses. That is, it tends to make the decision whether to group a new alert with an existing alert class less confident.

The ability to anticipate the next stage in a staged attack is also achieved by a transition in state. Internally, a transition matrix is maintained that expresses the probability of the next state in a staged attack, given the present state. Persistence (the fact that the next step may match the current) is explicitly captured. The transition gives a prior distribution over attack states that is assumed to hold immediately before the next alert is analyzed.

After the transition models generate new prior belief states for their respective alert classes, the similarity expectation for each feature in a new alert is updated. The similarity of the new alert to all existing alert classes is then computed. If none match

above some similarity threshold, the new alert is defined to be a new alert class. Otherwise, it is assumed that the new alert is a continuation of the best-matching alert class. By including a list of contributing alerts in the alert class structure, full details of the new alert are retained.

E. Alert Correlation Method

Figure 4 is a flowchart showing how alerts may be aggregated into classes of related alerts in a preferred embodiment. As was discussed above, both alerts and alert classes each have one or more distinguishing features that can be assigned different values. First, a set of potentially similar features shared by a new alert and one or more existing alert classes is identified (step 401). Next, an expectation of similarity between the features of the new alert and features of one or more existing alert classes is either generated or updated (step 403). After a comparison between the new alert and the existing alert class(es) is complete (step 405), the new alert is either associated with the existing alert class that it most closely matches (step 407), or new alert class is defined to include the new alert (step 409).

Although preferred embodiments have been discussed above, the scope of this invention is defined by the following claims: